Chapter 7
Typical Mistakes in MDS

Abstract Various mistakes that users tend to make when using MDS are discussed, from using MDS for the wrong type of data, using MDS programs with suboptimal specifications, to misinterpreting MDS solutions.

Keywords Global optimum · Local optimum · Termination criterion
Initial configuration · Degenerate solution · Dimensional interpretation
Regional interpretation · Procrustean transformation

7.1 Assigning the Wrong Polarity to Proximities

A frequent beginner's mistake is scaling proximities with the wrong polarity. If the data are similarities, but MDS treats them as dissimilarities (or vice versa), it will generate a misleading solution with very high Stress. The MDS program cannot know how to interpret the data and, therefore, works with its default interpretation of the data. This usually means that the data are taken as dissimilarities. Yet, correlations, for example, are similarities, because greater correlation coefficients indicate higher similarity and, therefore, they should be represented by relatively small distances. If the user incorrectly specifies the data's polarity, then MDS cannot generate meaningful solutions.

7.2 Using Too Few Iterations

Many MDS programs have suboptimal default specifications. In particular, they typically terminate the iterations of their optimization algorithms before the process has actually converged at a local minimum. This premature termination is caused by setting the termination criteria too defensively. Many programs set the maximum number of iterations to 100 or less, a specification that dates back to the times when computing was slow and expensive. For example, the GUI box of SYSTAT in Fig. 1.5 shows that, per default, this MDS program allows at most 50 iterations. The iterations

© The Author(s) 2018

I. Borg et al., *Applied Multidimensional Scaling and Unfolding*,
SpringerBriefs in Statistics, https://doi.org/10.1007/978-3-319-73471-2_7

are also stopped if the Stress does not go down by more than 0.005 per iteration. However, one can show that very small Stress reductions do not always mean that all points remain essentially fixed in further iterations. We therefore recommend to always *clearly* change these default values to allow the program to work longer. Instead of a maximum of 50 one can easily require 1,000 or more iterations. The convergence criterion, in turn, could be set to 0.000001 or smaller, i.e., to a very small value indeed.

7.3 Using the Wrong Initial Configuration

All MDS programs automatically generate their own initial configuration if the user does not provide an external starting configuration. It is a common fallacy to assume that internally generated starting configurations will always lead to optimal MDS solutions. For example, we have found in many tests that the default starting configuration used in PROXSCAL (called "SIMPLEX") is often not optimal. We recommend using the option INITIAL=TORGERSON instead. Yet, *no* starting configuration—rational or user-provided—always guarantees the best-possible final solution, and so the user should test some sensible alternatives before accepting a particular MDS solution all too early as the final solution.

Random starting configurations can also be useful in MDS. Indeed, *many* random configurations can easily be used without much effort. For example, for the solution in Fig. 1.4 we used PROXSCAL with the option RANDOM=1000; i.e., we asked the program to repeat the scaling with 1,000 different random starting configurations and then report the solution with the lowest Stress value. That only took seconds with this small data set.

The same method can also be used with `mds()` in SMACOF. However, `mds()` generates only 1 random configuration when setting the argument `init="random"` in `mds()`. Thus, we have to program a loop to find the best solution or use the function `random.multistart()` below (which here calls an ordinal MDS and 500 random starts):

```
diss <- sim2diss(wish, method=7)
set.seed(123)
random.multistart <- function(diss, type="ordinal", nrep=100) { s1 <- 1
  for (i in 1:nrep) { out <- mds(diss, type=type, init="random")
  if (out$stress < s1) { object <- out; s1 <- out$stress }}
return(object) }
result <- random.multistart(diss, type="ordinal", nrep=500)
result
```

Running the above commands leads to a `result$stress` of .185 for the country similarity data from Sect. 2.2. Repeating this analysis with different seeds leads to the same minimum Stress value in each case. So, .185 seems to be the best-possible Stress for ordinal MDS of these data.

Sometimes there exist several *different* solutions that all have *almost* the same small Stress value. In that case, the user can pick the solution that is most convincing in terms of interpretation. The problem is that all MDS computer programs only report the best solution they found, where “best” obviously only says that it has the smallest Stress. No program can consider a configuration’s meaning as an additional criterion. To help finding possible solutions that have both an acceptable Stress but differ in their substantive meaningfulness, Borg and Mair (2017) suggest a strategy where all MDS solutions that result from many different initial configurations are stored and then compared with respect to their structural similarity. This strategy is implemented in the `icExplore()` function. It generates a large set of MDS solutions using random initial configurations, matches them all by Procrustean fittings, computes the inter-correlations of their point coordinates, and finally runs an (interval) MDS of these inter-correlations.

```
diss <- sim2diss(wish, method=7)
set.seed(3)
solutions <- icExplore(diss, type="ordinal", nrep=75)
solutions
plot(solutions)
```

The result of this analysis for the country similarity data using 75 random initial configurations is shown in Fig. 7.1. The numbers in the plot represent the MDS configurations, and the size of the numbers corresponds to the Stress of the solution (solution #64, thus, has a poor fit to the data). The distances among the points represent the similarities of the configurations. The plot thus shows that there are many different local minima solutions when random initial configurations are used. Many of these solutions have a poor fit, but there are two clusters of highly similar configurations on the right-hand side that all have relatively low Stress. The user can take a look at, say, #9 (in the upper cluster on the right-hand side of Fig. 7.1) and #25 (in the cluster underneath) to see how they differ and which one is better interpretable (see discussion in Sect. 7.8). One can plot #9, say, by simply calling `plot(solutions[[9]])`. The Stress is printed by `solutions[[9]]`.

The user can also follow another strategy. Compute a Stress-optimal MDS solution first, study its interpretability, and then possibly move some points “by hand” to theoretically more pleasing positions. These hand movements can be translated into changes of the coordinates of these points. The modified coordinate matrix can subsequently be used as the initial configuration in `Stress0()`. This function computes the Stress of the modified solution (without any iterations). Alternatively, one may set `niter=1000`, for example, and hope that the program will find an optimal solution with an acceptably small Stress that lies in the vicinity of the modified configuration.

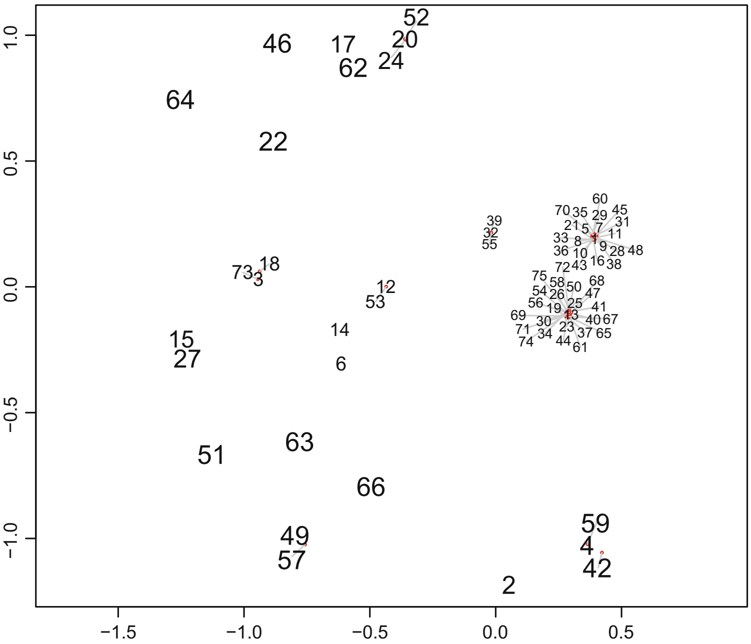

Fig. 7.1 Similarity structure of MDS solutions based on 75 random initial configurations for country similarity data; number represents solution; size of number represents Stress of solution

Finally, a theory-generated initial configuration (if it can be derived) is always a choice that should at least be tested. Consider, for example, the data on the similarity of rectangles and the data on personal values discussed in Chap. 2. In both cases, there were clear hypotheses about the expected MDS structure of the data. These predictions can easily be translated into coordinate matrices that then serve to define initial configurations. For example, for the rectangle data, one can simply read off the coordinates from Fig. 2.4 or call `data(rect_constr); S <- rect_constr` and then tell the MDS program to use **S** as an initial configuration.

There is usually no need to formulate the initial configuration as precisely as in case of matrix **S** above, nor does the theory always allow such precise predictions. This is certainly true for the personal values data, where the theory predicts a circle with points ordered as PO - AC - HE - ST - SD - UN - BE - TR - CO - SE - PO. No prediction can be derived for the distances among the points on the circle and so one could spread them out evenly, for example. It suffices to plot this configuration on a piece of paper, co-ordinatize its points by a simple grid, and then coarsely read off these coordinates to generate a matrix like **S** above. Of course, one could also do this on the computer screen, then plot the coordinate matrix to visually check it, and possibly adjust the point coordinates repeatedly until the configuration seems right.

7.4 Doing Nothing to Avoid Suboptimal Local Minima

MDS always tries to find the local minimum solution with the smallest possible Stress, i.e., the *global minimum*. MDS users can do their share to help find this global minimum by keeping an eye on the following issues:

- A good initial configuration is the best way to avoid suboptimal local minima. If you have a theory, then a user-defined configuration is what you should always use. If you do not have a theory, you must leave it to the MDS program to define its own starting configuration. In that case, we recommend using the solution of classical MDS (also known as the *Torgerson solution*) as a start which is indeed the default initial configuration of `mds()` in the SMACOF package.
- Another precaution against suboptimal local minima is using multiple random starts. As modern MDS programs are extremely fast, one can easily require the program to repeat the scaling with a very large number of different random starts (e.g., with 1,000 or more).
- City-block distances increase the risk to end up in suboptimal local minima. General MDS programs are particularly sensitive in this regard. There exist MDS programs that are optimized for city-block distances, but they are hard to obtain and typically require expert support for using them.
- The greater the dimensionality of the MDS space, the smaller the risk for suboptimal local minima. The main problem in low-dimensional spaces (1d, in particular) is that swapping points in space by iteratively repeating small movements is difficult, because such movements may first increase the Stress before it goes down. Hence, even if you want, say, a two-dimensional MDS solution, using the first two principal components of a three-dimensional MDS solution may serve as a good initial configuration.
- Suboptimal local minima are particularly likely in case of one-dimensional MDS. Standard programs almost never find the global minimum. If you must do one-dimensional MDS, you should provide an external starting configuration computed with 2d MDS (see above), or use an MDS program for the 1d case. Special 1d MDS programs are based on permutation algorithms which are computationally demanding. An example is `uniscale()` in the SMACOF package: It finds the permutation of the points with the smallest Stress, but always assumes that the data are on a ratio scale. Yet, one may use its solution as an initial configuration in ordinal and interval MDS.

7.5 Not Recognizing Degenerate Solutions

Of all MDS models, ordinal MDS is the model that has been used most often. It allows any rescaling of the data that preserves their order, but it nevertheless produces stable metric solutions. However, ordinal MDS can run into a special problem that the user should keep an eye on; i.e., it can lead to *degenerate* solutions. Consider the following example. Table 7.1 exhibits the inter-correlations of eight test items of

Table 7.1 Correlations (lower half) of some test items of the KIPT and their ranks (upper half).

	NP	LVP	SVP	CCP	NR	SLP	CCR	ILR
Nonsense word production (NP)	–	9	4	1	6	19	10	12
Long vowel production (LVP)	.78	–	1	7	5	21	20	22
Short vowel production (SVP)	.87	.94	–	3	2	17	16	23
Consonant cluster production (CCP)	.94	.83	.90	–	7	14	11	16
Nonsense word recognition (NR)	.84	.85	.91	.83	–	17	15	18
Single letter production (SLP)	.53	.47	.56	.60	.56	–	13	16
Consonant cluster recognition(CCR)	.72	.48	.57	.69	.59	.62	–	8
Initial letter recognition (ILR)	.66	.45	.44	.57	.55	.57	.82	–

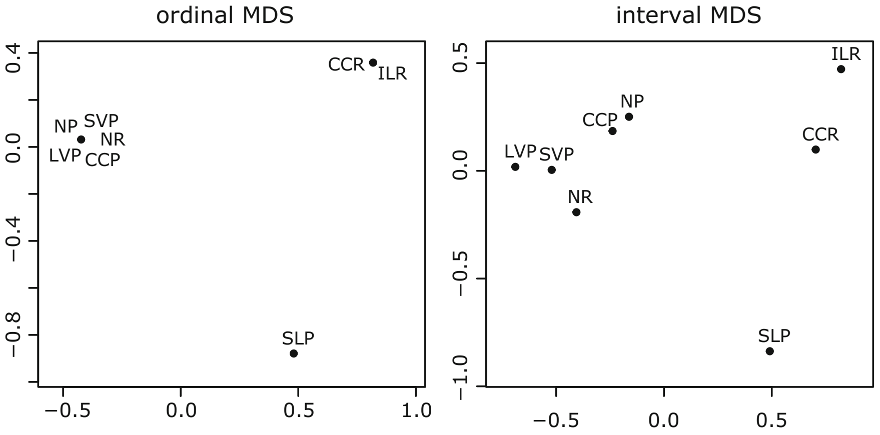

Fig. 7.2 Ordinal and interval MDS representations for data of Table 7.1

the Kennedy Institute Phonics Test (KIPT), a test for reading skills (Guthrie 1973). If we scale these data by ordinal MDS using `mds()` (see commands below[1]), we obtain the configuration in Fig. 7.2 (left panel). Its Stress value is zero, so this MDS solution is formally perfect. Yet, the Shepard diagram of this solution (see left panel of Fig. 7.3) reveals a peculiar relation of data and distances: Although the data scatter evenly over the interval from .44 to .94, they are not represented by distances with a similar distribution, but rather by two clearly distinct classes of distances so that the regression line makes just one big step.

```
diss <- sim2diss(KIPT)
fit1 <- mds(diss, type="ordinal", eps=1e-11)
fit2 <- mds(diss, type="interval")
fit3 <- mds(diss, type="ratio")
```

[1] Note that we set the argument `eps` to an extra-small value here to make the program iterate on and on until it reaches such an exotically small raw Stress value if it can be reached in `itmax=3333` iterations. Without this argument, `mds()` will use the default value `eps=1e-06` which causes it to stop earlier.

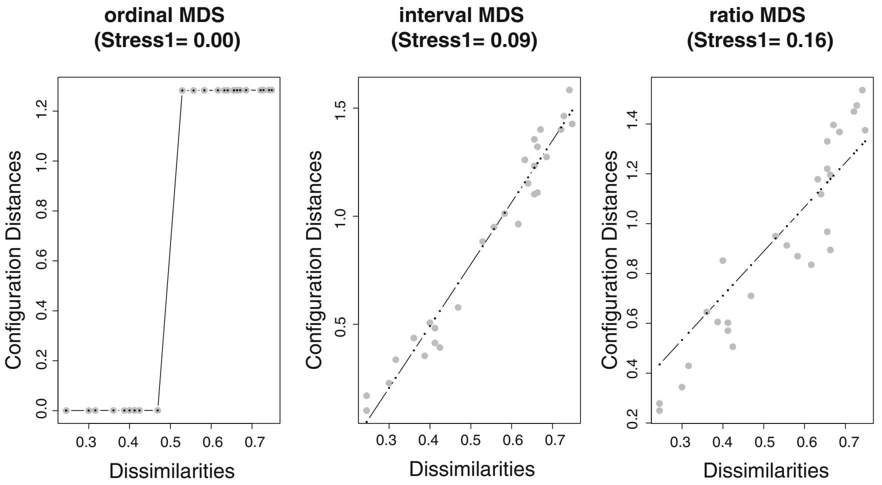

Fig. 7.3 Shepard diagrams for ordinal MDS, interval MDS, and ratio MDS of the correlations of Table 7.1 (converted into dissimilarities)

The MDS configuration shows three clusters that form an equilateral triangle. This configuration represents all large correlations ($r \geq .78$) by distances close to zero and all smaller correlations ($r < .72$) by the same large distance. This solution correctly displays a few data relations, but loses whatever else there is in the data. The perfect Stress value is, therefore, deceptive. The large and the small distances, respectively, *can be reordered arbitrarily* as long as all similarities within the blocks marked in Table 7.1 remain greater than all between-block similarities. Any such reordering will have no effect on the Stress value.

The reason for such a degenerate solution is that the data have a peculiar structure. They form three subgroups, with high within- and low between-correlations. With ordinal MDS, such data can always be scaled with zero Stress. Of course, the data here are particularly selected to demonstrate degeneracy. In practice, one should rarely find such cases, but the problem becomes more likely if the number of variables is small ($n \leq 8$).

If the Shepard diagram suggests that the MDS solution is degenerate, then the natural next step for the user is testing a *stronger* MDS model and comparing the solutions. Using interval MDS with the above data yields the solution in the right panel of Fig. 7.2. It too shows the three clusters of test items, but it does not collapse them. Its Shepard diagram (see the middle panel of Fig. 7.3) makes clear that the interval solution preserves a *linear* relationship of the data in Table 7.1 to the distances in Fig. 7.2.[2]

[2] Note that if you plot the correlations of Table 7.1 rather than the dissimilarities on the Y-axis of the Shepard diagram of the interval MDS—using `plot(aus1, plot.type="Shepard", shepard.x=kipt)`—the regression line is slightly curved. This is so because transforming the correlations into dissimilarities via $\delta_{ij} = \sqrt{1 - r_{ij}}$—which is what

An even stronger model is ratio MDS. For these data, however, it is too strong. The Shepard diagram (right panel of Fig. 7.3) shows that it not only drives up the Stress, but it does so causing a *systematic* error: Small distances are almost all too small, and very large distances are too large (see the scatter of the points about the regression line in the Shepard diagram). These errors are the consequence of insisting that the regression line must run through the origin (0.00, 0.00).

7.6 Meaningless Comparisons of Different MDS Solutions

A frequent issue in MDS applications is comparing two or more MDS solutions. Consider a study by Borg and Braun (1996). They were interested in the difference between East Germans and West Germans in their work values shortly after Germany reunited in 1990. The items asked the respondents to rate 13 aspects of their work life (such as “high income” or “good chances for advancement”) on a scale from “not important” to “very important” to them personally. Scaling the inter-correlations of the two samples leads to two-dimensional MDS solutions, but even though they have just 13 points each, they are difficult to compare, because one must ignore *meaningless* differences that are due to different orientations of the plots. It is like comparing two maps of different size, and one is upside down, for example. When comparing MDS plots, one can eliminate such meaningless differences optimally by *Procrustean transformations*. If configuration **X** is taken as the target, the other configuration **Y** is rotated, reflected, translated, and adjusted in its size to optimally match **X**. All these transformations are *similarity transformations* that do not change the structure of the MDS configurations. Differences between two configurations[3] that can be eliminated by similarity transformations cannot possibly be meaningful, because they are *not caused by the data*. We do apply this method for the East and West German MDS configurations using these commands:

```
labels.short <- c("interesting","independent","responsibility","meaningful",
"advancement","recognition","help others","useful","social","secure job",
"income", "spare time", "healthy")
attr(EW_eng$west, "Labels") <- attr(EW_eng$east, "Labels") <- labels.short
res.west <- mds(sim2diss(EW_eng$west, method="corr"), type="ordinal")
res.east <- mds(sim2diss(EW_eng$east, method="corr"), type="ordinal",
init=res.west$conf) ## note the initial configuration here
fit2 <- Procrustes(res.west$conf, res.east$conf)
plot(fit2)
## compute overall similarity measures: r and c
r <- cor(as.vector(res.west$conf), as.vector(fit2$Yhat))
c <- fit2$congcoef ## congruence coefficient on distances
```

`diss <- sim2diss(kipt, method="corr")` is doing—is a slightly nonlinear function. This is irrelevant for ordinal MDS, but it shows up in interval MDS.

[3] Procrustean fittings can also be used for configurations that differ in the number of points and in their dimensionalities. For example, the configuration in Fig. 2.13 was fitted to the configuration in Fig. 2.12 to make comparisons easier. The target **X** was derived from Fig. 2.12 by roughly reading off the X- and Y-coordinates of the centroids of the various value groups. In case of different dimensionalities, one can simply add column vectors with only zeroes to **X** or to **Y**.

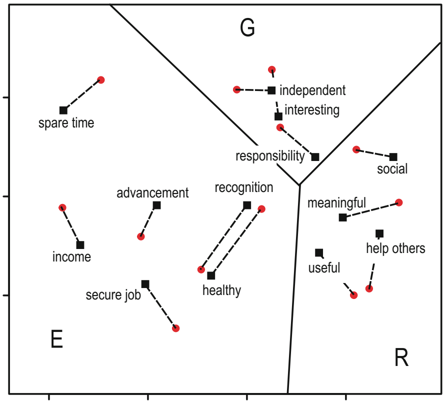

Fig. 7.4 Overlay plot of West German (squares) and East German (dots) work value configurations, optimally matched, with partition based on the ERG theory

The plot in Fig. 7.4 shows the East and the West German results, optimally fitted to each other in one overlay plot. To measure the similarity of the configurations, one can compute the congruence coefficient of corresponding distances ($c = .964$) or the correlation of the coordinates of corresponding points (after Procrustean fitting: $r = .914$). These coefficients can be evaluated against the fit of random configurations (see R script below which yields benchmark values of .88 and .62 for the c- and the r-coefficients, respectively). Hence, the similarity of the observed configurations is much higher than can reasonably be expected by chance.

```
Procrustes.test <- function(n,m,nrep=500) { set.seed(333); c <- vector()
r <- vector(); X <- matrix(runif(n*m, -1, 1), nrow=n,ncol=m)
X <- scale(X, scale=FALSE)
for (i in 1:nrep) { Y <- matrix(runif(n*m, -1, 1), nrow=n, ncol=m)
fit <- Procrustes(X, Y); c[i] <- fit$congcoef
r[i] <- cor(c(X), c(fit$Yhat))}
cr <- list("c"=c, "r"=r) }; z <- Procrustes.test(13,2) ## 13 points in 2d
z99 <- quantile(z$c, .99); r99 <- quantile(z$r, .99) ## 99% quantiles
cat("c(99%)=", round(z99,2), " r(99%)=", round(r99,2), sep = '')
```

Apart from their significant point-to-point similarity, one here notes that both configurations can be partitioned in the same way by Alderfer's E(xistence), R(elations), and G(rowth) theory (Alderfer 1972a). This is a higher-order form of similarity, and it may hold even if the point-wise correspondence is not that high.

7.7 Evaluating Stress Blindly

A frequent mistake of MDS users is that they are often too quick in rejecting an MDS solution because its Stress seems too high. The Stress value is, however, merely a *technical* index, a target criterion for an optimization algorithm. An MDS solution can be robust and replicable, even if its Stress value is high. Stress, moreover, is *substantively blind*; i.e., it says nothing about the compatibility of a content theory with the MDS configuration, or about its interpretability.

Stress is a *summative* index for *all* proximities. It does not inform the user how well a *particular* proximity value is represented in the given MDS space. This was discussed in detail in Chap. 3. The least one can do is to take a look at the Stress-per-point values. Unfortunately, not all MDS programs compute SPP values (or similar point-fit measures). However, most programs allow saving the configuration's distances so that one can compute appropriate point-fit measures with standard data analysis programs (e.g., the correlation between the proximities and the corresponding MDS distances).

A simple way to deal with ill-fitting points is to eliminate them from the analysis. This popular approach is based on the rationale that such points have a special relation to the other points that needs additional considerations. Another solution is to increase the dimensionality of the space so that these points can move into the extra space and form new distances. The rationale in this case is that the proximity of the objects represented by these points to the other points is based on additional dimensions that are not relevant in other comparisons. Experience shows, though, that SPP values are often quite unstable. For example, SPP plots change a lot under different MDS models so that "special" points cannot always be identified with confidence.

In any case, accepting or rejecting an MDS representation on the basis of overall Stress can be too simple. This is easy to see from an example. Consider the West German MDS configuration in Fig. 7.4. If we increase the dimensionality of this solution to $m = 3$, the Stress goes down from 0.17 to 0.09. If we proceed in the same way in case of Fig. 2.2, we get the same reduction in Stress. However, in the former case, the reduction in Stress is caused by essentially two points only. That is, "healthy working conditions" and, in particular, "(much) spare time" clearly move out of the plane in Fig. 7.4 into the third dimension. In case of the country similarity data, all points jitter (some more, some less) about the plane, which looks as if the third dimension is capturing essentially only noise.

For data with large noise components, therefore, low-dimensional MDS solutions can have high Stress values, but they may still be better in terms of theory and replicability than higher-dimensional solutions with lower Stress values. In that case, a low-dimensional solution may be an effective data smoother that brings out the true structure of the data more clearly than an over-fitted high-dimensional MDS representation.

7.8 Always Interpreting Principal Axes Dimensions

Interpreting an MDS solution can be understood as projecting given or conjectured content knowledge onto the MDS configuration. The country similarity example of Sect. 2.2 demonstrates how this is typically done: What one interprets are *dimensions*. MDS users often *automatically* ask for the meaning of "the" dimensions, by which they often mean the axes of the plot that the MDS program delivers. These axes are almost always the principal axes of the solution space. Yet, this dimension system can be arbitrarily rotated and reflected, and oblique dimensions would also span the plane. Hence, users do not have to interpret the dimensions offered by the MDS program, but they could look for m dimensions (in m-dimensional space) that are more meaningful.

There is, however, no natural law that guarantees that dimensions are meaningful at all. Thus, one should be open for other ways of interpreting MDS solutions. One possibility is to look for meaningful *directions* rather than for dimensions. A direction corresponds to a simple line that runs through the MDS plot. When projecting the points of the configuration onto such a line, it becomes an *internal scale*. One can plot such internal scales through a common point such as the centroid of the configuration. Points to the left of this anchor point are given negative scale values; those to the right of it receive positive values. To interpret the internal scale, one studies the point distribution with a focus on content questions such as these: What points lie at the extremes of the scale? How do they differ in terms of content? What is the attribute where they differ most? Why are the points $i, j, \ldots$ so close together? What do they have in common? Answering such questions gives meaning to the scale.

Additional data can be helpful in such interpretations. We show this for the country similarity example. Table 2.1 exhibits the coordinates of the MDS solution in Fig. 2.2 and the countries' values on two *external* scales, economic development, and number of inhabitants. These scales can be fitted into the MDS space by using the `mdsbiplot()` function as follows, yielding Fig. 7.5. The fit of the external scales in this MDS configuration is given by the correlation of these scales with the projections of the points onto straight lines through the arrows that represent them. We here get $r = .94$ for economic development and $r = .46$ for the number of inhabitants. (The length of the two arrows represents, approximately, the relative fit of the external scales.) This suggests to interpret this solution in terms of a rotated set of dimensions that correspond to the two arrows representing economic development and number of inhabitants.

```
diss <- sim2diss(wish, method=7)
res <- mds(diss, type="ordinal")
ecdev <- c(3,1,3,3,8,3,7,9,4,7,10,6)
inhabs <- c(87,17,8,30,51,500,3,100,750,235,201,20)
labs <- attr(wish, "Labels")
fitbi <- biplotmds(res, cbind(ecdev, inhabs))
plot(fitbi, main="", xlab="", ylab="", cex=1.3,
     label.conf=list(cex=1.2, pos=ifelse(labs!="RUSSIA", 3, 1)),
     vecscale=0.5, vec.conf=list(cex=1.2, col="red", cex=1.2, length=0.1))
```

External scales can also help in choosing among different MDS solutions with almost the same Stress. For the country similarity data, ordinal MDS starting with different random configurations leads to a set of different solutions (see Fig. 7.1). Many of them have unacceptably high Stress, but there are different solutions (e.g., #1 and #13) with the same minimal Stress of .185. Figure 7.6 shows these solutions next to each other. In each solution, the two external scales were fitted into the configurations by multiple regression (as in Fig. 2.16, for example).

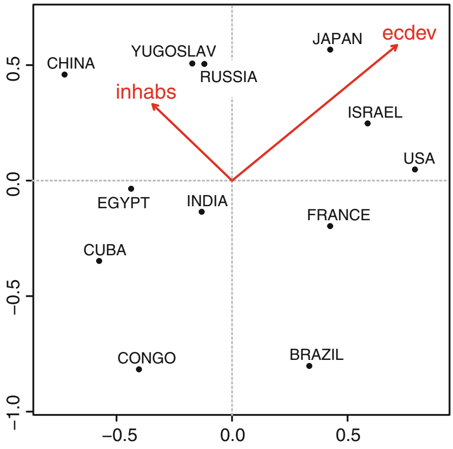

Fig. 7.5 MDS solution for country similarity data; fitted external scales shown as arrows

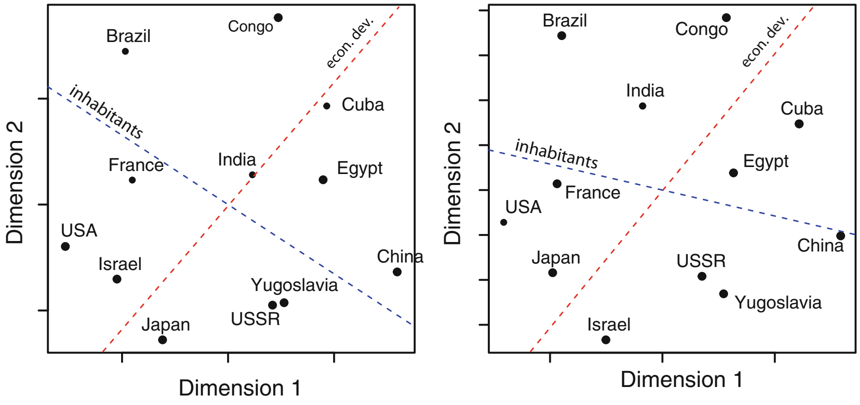

Fig. 7.6 Two same-Stress MDS solutions for country similarity data, with fitted external scales

The two solutions are rather similar (after Procrustean fitting) but differ in two important details: In the left configuration, the positions of Japan and Israel are swapped in comparison with where they are in the right configuration; moreover, in the left configuration, India is positioned more in the center of the configuration. This means that in the configuration on the left, the very large countries are closer together on the line “inhabitants.” So, this internal scale correlates with the external scale “number of inhabitants” (see Table 2.1) with $r = .46$ in the left configuration, but only with $r = .30$ in the right configuration. At the same time, the fitted external scales correlate with $r = .93$ in both plots. Hence, the configuration on the left is the somewhat more meaningful MDS solution if one wants to follow Wish (1971) in interpreting the configuration in terms of these dimensions. However, this solution may not be the one that is reported by the MDS program as the final solution, but you can find it if you use a proper initial configuration identified by `icExplore()`.

7.9 Always Interpreting Dimensions or Directions

Dimensions and, more generally, directions are but special cases of *regions*. Regions are subsets of points of an MDS space that are *connected* (i.e., each pair of points in a region can be joined by a curve whose points lie completely within this region), *non-overlapping*, and *exhaustive* (i.e., each point lies in exactly one region). When interpreting MDS solutions, we ask to what extent certain classifications of the objects on the basis of *content facets* correspond to regions of the MDS space. Expressed differently, we ask whether the MDS configuration can be *partitioned* into substantively meaningful regions and, if so, how these regions can be described.

An example for such a partitioning is shown in Fig. 7.4. Here, the different objects ("work values") were first classified into three categories on the basis of a theory by Alderfer (1972b): Work values related to outcomes that satisfy existential-material needs (E), social-relational needs (R), or cognitive-growth needs (G). This ERG typology surfaces in MDS space in certain neighborhoods that can be separated from each other by cutting the plane in a wedge-like fashion. The same type of partitioning is possible both in the West German and also in the East German MDS plane. Hence, the two solutions are equivalent in the ERG sense (Borg and Braun 1996).

Partitioning an MDS space is done *facet by facet*. For each facet F_i, one generates a *facet diagram*. This is simply a copy of the MDS configuration where each point is replaced by the code that indicates to which category of F_i the respective point belongs. One then checks to what extent and in which way this facet diagram can be partitioned into regions that contain only codes of one particular type. The emerging regions should be as simple as possible, e.g. with straight partitioning lines. This is desirable because simple partitions can also be characterized by simple laws of formation that promise to be more robust and more replicable than complicated patterns that are fitted too closely to the particular data and its noise.

Although there exist computer programs that yield partitions for facet diagrams that are optimal in some sense (Borg and Shye 1995), it is typically more fruitful for the user to work with pencil and eraser on a printout of the facet diagram. This way, partitioning lines can be drawn, redrawn, and simplified in an open-eyed fashion, paying attention to content and substantive theory. One may decide, for example, to admit some placements of points in "wrong" regions, because simple overall patterns with some errors are better than perfect partitions with overly complicated partitions.

Three prototypical regionalities that often arise in practice are shown in Fig. 7.7: *axial*, *modular*, and *polar* partitions. Axial and modular partitions are either based on ordered facets, or they suggest ordered facets. Polar partitions, in contrast, are typically related to unordered (nominal) facets. Of course, if the sectors in a polar partition are arranged similarly in many replications, then one should think about reasons for this order.

Regionalizations—simple ones, in particular—become unlikely to result by chance if the number of points goes up. That is easy to see from a thought experiment. Assume you take a set of *n* ping-pong balls and label some of them with "a", others

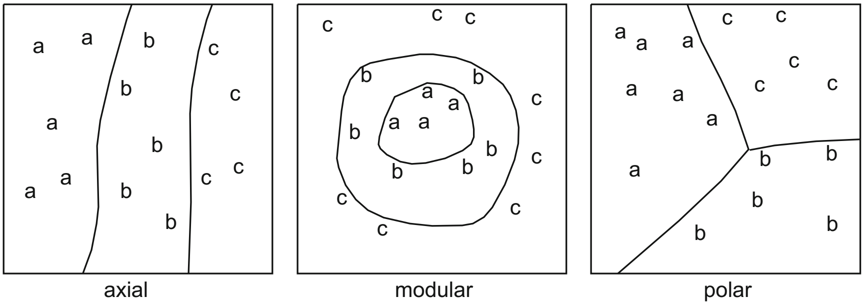

Fig. 7.7 Prototypical partitioning of MDS configurations by three facets, each one with three categories (a, b, c)

with "b", and still others with "c". Then, throw them all into a bucket, mix them thoroughly, and pour the bucket onto the floor. After the balls come to their parking positions, try to partition the resulting configuration into a-, b-, and c-regions. This will be difficult or even impossible if you want simple regions as in Fig. 7.7. It is even less likely that the regionality that you find in one case can be replicated when the experiment is repeated. A simple regional pattern, therefore, suggests a lawful relationship in the sense that *the facet structures the observations*. This notion becomes even more powerful if an MDS configuration can be partitioned by more than one facet so that the different organizational patterns can be stacked on top of each other as, for example, in the radex in Fig. 2.8.

An MDS solution can be partitioned, in principle, by as many facets as the user can think of. There is no fixed relation between the number of facets and the dimensionality of the space. This is different for dimensions: In an m-dimensional space, one always seeks to interpret exactly m dimensions. A dimensional interpretation corresponds to a combination of m axial facets (see Fig. 7.7, left panel), each generating an ordered set of (infinitely) narrow bands with linear boundary lines so that a grid-like mesh (as, e.g., in Fig. 6.3) is generated.

Regions are sometimes confused with *clusters*. Clusters, however, are but special cases of regions. They are often defined as lumps (or chains) of points surrounded by empty space so that each point in a cluster is always closer to at least one point in the cluster than to any point not in the cluster. Clustering in that sense is not required for perfect regions. Regions are like countries that cut a continent like Europe into pieces. Malmö/Sweden, for example, is much closer to Copenhagen/Denmark—both are connected by a bridge—than to any other Swedish city, so the Swedish cities do not form a cluster on the European map, but they are all in the same region.

Clusters, moreover, are *formal* constructs, while regions are based on *substantive* thinking that is often expressed via facets. Nevertheless, one can always cluster proximities and then check how the resulting clusters organize the points of an MDS solution. Cluster analysis is, however, not particularly robust: Different amalgamation criteria can lead to vastly different clusters. Cluster analysis, therefore, is not a method for "validating" an MDS solution or interpretation, as some writers argue.

Rather, cluster analysis typically just leads to groupings of points that tend to surface similarly in MDS solutions.

7.10 Poorly Dealing with Disturbing Points

A frequent problem in MDS applications is what to do with points that do not fit into an interpretation. A typical case is a configuration that cannot be partitioned in a theoretically pleasing way because of one or a few “misplaced” points. In such cases, one may decide to construct (slightly) overlapping regions, or stick to the partitioning notion and generate curvy partitioning lines (as, e.g., in Fig. 6.2). A third solution is to draw a best-possible partitioning system where some points remain in regions to which they do not belong. A fourth, and often rather dubious solution, is to eliminate such points from the MDS configuration by “explaining them away” in substantive terms.

A completely different way to deal with disturbing points is asking how much the Stress goes up if one shifts these points in space such that simple partitioning becomes possible. The easiest way to answer this question is the following. Assume you use `res <- mds(diss, type="interval")`. Now, replace the coordinates of disturbing points in `res$conf` with “should” coordinates (i.e., coordinates that put these points into positions where they are not disturbing anymore). Let us call this modified coordinate matrix `X.mod`. Then, compute the Stress of `X.mod` using the `stress0()` function: `stress0(diss, init=X.mod, type="interval")`. Finally, compare the Stress value of the optimal solution `res$conf` with the Stress of `X.mod`. If the Stress increment is small, then one would probably prefer the solution that allows a simple interpretation over the optimal-Stress solution. The rationale is that it promises to be better replicable, being based on a substantive law of formation, than the solution that represents the one given set of data with minimal Stress.

A formally better solution is using confirmatory MDS. However, confirmatory MDS with regional restrictions can be difficult to formulate and to implement. Hence, before trying this, a simple shift-and-see approach yields a quick answer that is often sufficient. Note, though, that replications are *absolutely essential* in any case. If certain disturbing points come out similarly in replications, one must take a closer look at what exactly is being measured by them and how this is related to the rest of the variables. A small increment in global Stress when shifting a few points can also be *deceptive*, in particular if only one or two points are moved and the rest of a large configuration is not changed. A vivid example is the case of the Morse signals in Fig. 6.3, where only one point (the signal for “1”) is substantially shifted out of a total of 36 points. This one-point movement cannot affect the Stress very much and so this one signal remains suspicious.

7.11 Scaling Almost-Equal Proximities

Proximity data cannot always be represented in a low-dimensional space. This is true, for example, if the data have a large error component or if they are simply random data. A second instance is data that are essentially constant. Buja et al. (1994) have shown that if all data are exactly equal, 2d ratio MDS leads to points that all lie on concentric circles; moreover, the points can all be interchanged without affecting the Stress. Users should therefore keep an eye on the case of almost-equal proximities or disparities. In particular, they must look closely at the units of the Y-axis of the Shepard diagram: If most of these values are almost equal, then caution is needed. Most computer programs choose an origin for the Y axis that magnifies the range of the observed values. If the origin of Y in a Shepard plot is zero, then the almost-equal problem becomes obvious immediately. Also, investigate the distribution of the proximities or disparities, preferably in a histogram. If the histogram shows that the disparities are all close together and are much different from zero, then one can expect the 2d solution of concentric circles.

Another way to diagnose peculiarities in the data is scaling them with different MDS models. In case of almost-equal proximities, ordinal MDS preserving ties (secondary approach) and interval MDS yield almost the same results. However, if ordinal MDS is used with the primary approach to ties—which allows to untie ties in the distances—a radically different solution is obtained, where most of the points cluster in one point, and a few points scatter about this cluster. The Stress, moreover, is much smaller than for the other MDS representations. If different MDS models yield such vastly different results, then something is almost always wrong. With well-structured data, different MDS models yield solutions that do not differ much.

7.12 Summary

Some mistakes are frequently made in MDS. One example is not specifying the proper polarity of proximities so that the MDS program uses similarity data as dissimilarity data, or vice versa. Another simple mistake is making MDS programs terminate their iterations too early, or not studying the effects of using different starting configurations. Once aware of these mistakes, they can be easily avoided. Another mistake is overlooking degenerate solutions in ordinal MDS. They can be avoided by using stronger MDS models. A rather frequent mistake is automatically asking for the meaning of "the" dimensions: Dimensions are but a special case of regions, and other meaningful patterns may also exist in an MDS configuration. Simply discarding disturbing points from an MDS solution is also too mechanical: Sometimes, such points can be shifted without affecting the Stress very much. Then, when comparing different MDS solutions, one should first get rid of meaningless differences via Procrustean transformations. Finally, data that are almost all equal can lead to meaningless MDS solutions.

References

Alderfer, C. P. (1972a). *Existence, relatedness, and growth*. New York: Free Press.

Alderfer, C. P. (1972b). *Existence, relatedness, and growth*. New York: Free Press.

Borg, I., & Braun, M. (1996). Work values in East and West Germany: Different weights but identical structures. *Journal of Organizational Behavior*, *17*, 541–555.

Borg, I., & Mair, P. (2017). The choice of initial configurations in multidimensional scaling: local minima, fit, and interpretability. *Austrian Journal of Statistics*, *46*, 19–32.

Borg, I., & Shye, S. (1995). *Facet theory: Form and content*. Newbury Park, CA: Sage.

Buja, A., Logan, B. F., Reeds, J. R., & Shepp, L. A. (1994). Inequalities and positive-definite functions arising from a problem in multidimensional scaling. *The Annals of Statistics*, *22*, 406–438.

Guthrie, J. T. (1973). Models of reading and reading disability. *Journal of Educational Psychology*, *65* 9–18.

Wish, M. (1971). Individual differences in perceptions and preferences among nations. In C. W. King & D. Tigert (Eds.), *Attitude research reaches new heights* (pp. 312–328). Chicago: American Marketing Association.